

Bixiv

纯前端项目，在1的基础上更新index.html和detail.html的js功能。

使用方式

使用服务器打开以避免跨域问题。

功能

index.html为主界面，detail.html为图片详细信息界面。

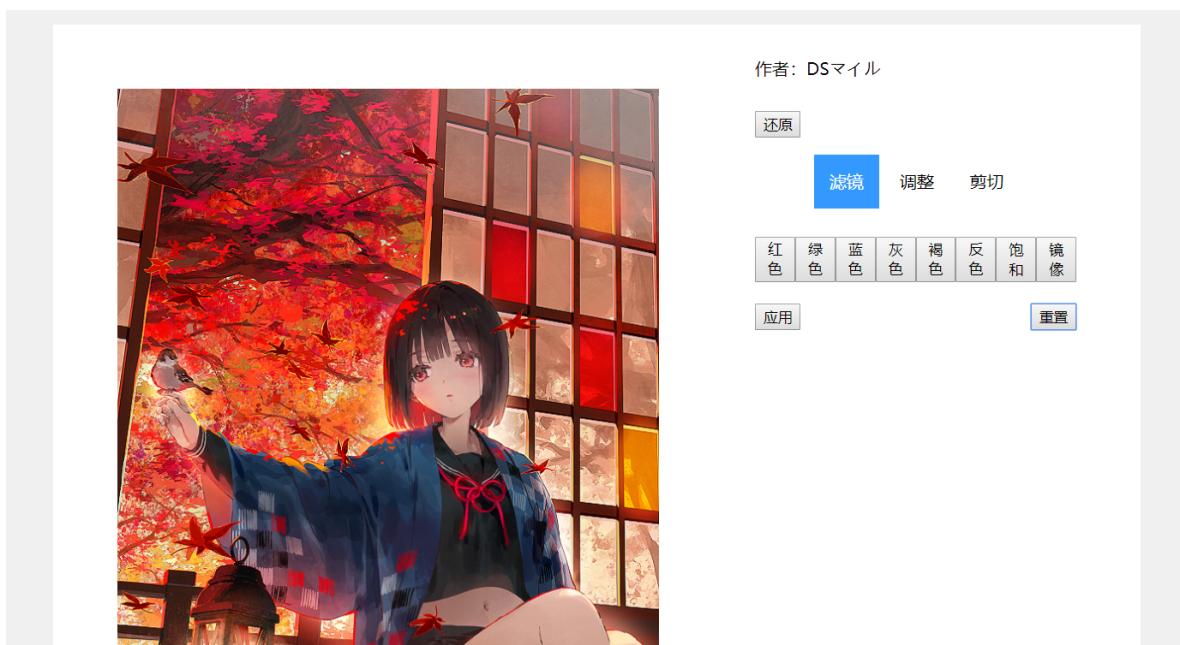
在主界面点击图片进入对应的detail界面，显示大图和图片相应的标题、作者等信息（时间有限未能加入每张图的信息）。

详细界面，可以在右侧菜单点击三个选项卡：滤镜、调整、剪切。

滤镜

显示了一排滤镜的按钮，点击其一可预览效果。点击应用将使当前的更改生效，点击重置将取消当前更改，选择其他选项卡将重置。

效果图-点击前：



效果图-点击后（褐色）：



作者: DSマイル

还原

滤镜 调整 剪切

红色 绿色 蓝色 灰色 褐色 反色 饱和 镜像

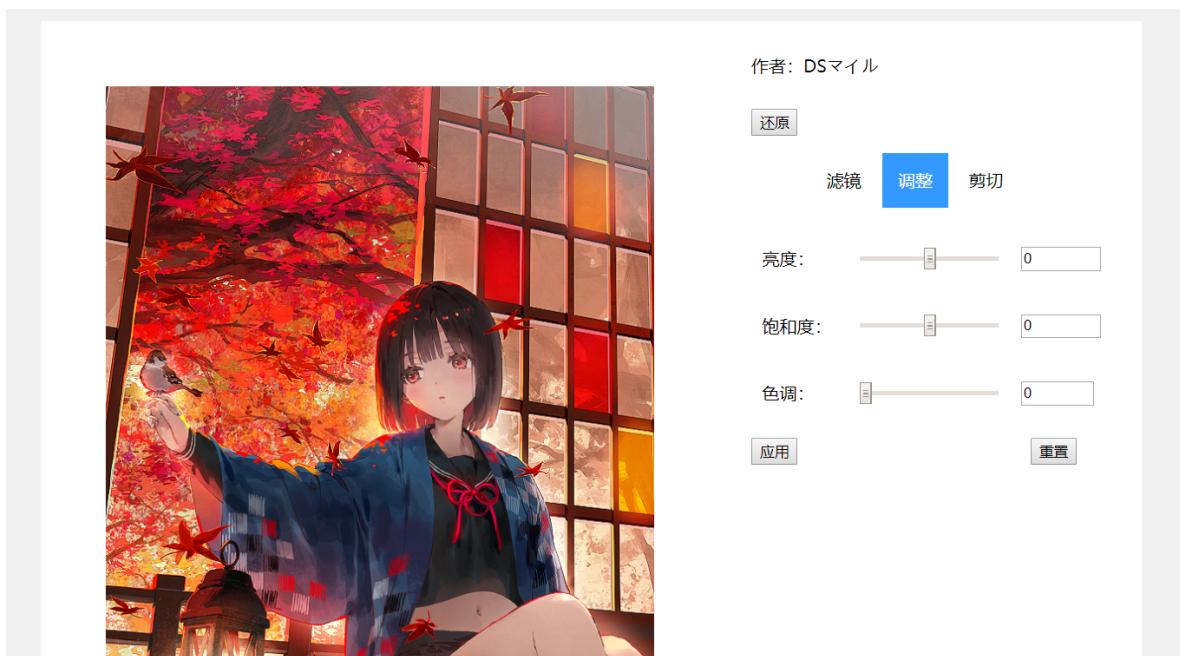
应用

重置

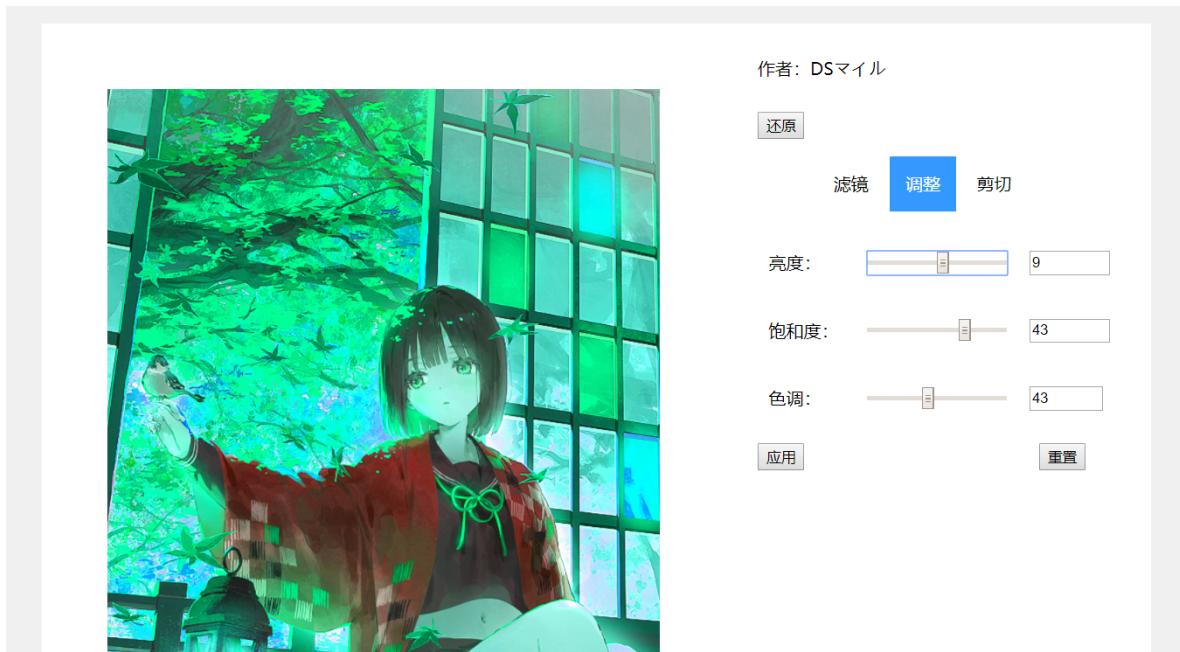
调整

可以调整图片的亮度、饱和度、色调。通过滑动条滑动或者输入数值完成后自动生成调整后的图片。点击应用将使当前的更改生效，点击重置将取消当前更改，选择其他选项卡将重置。

效果图-更改前：



效果图-更改后：



剪切

点击后自动出现可交互的剪切界面，可以在上面拖动、缩放选择框和图片。右边有预览图、逆时针旋转按钮、顺时针旋转按钮。点击应用将使当前的更改生效，点击重置将重置为初始状态，选择其他选项卡将退出剪切界面。

效果图：



代码实现

主界面

更新了index.html，使用jquery根据图片列表在界面上动态添加图片。

```

var i;
for(i=0; i<image_list.length;i++) {
    $(".list").append('<div class="image-container-fixed"> <a href="detail.html?source='
        + image_list[i]
        + '">  </a> </div>');
}
for(i=0; i<image_list.length;i++) {
    $(".container").append('<div class="image-container"> <a href="detail.html?source='
        + image_list[i]
        + '">  </a> </div>');
}

```

详细界面

更新了detail.html，添加基础功能和滤镜、调整、剪切三个功能。

传参功能

url传参获取当前的图片。

```

function getUrlParam(name) {
    var reg = new RegExp("(^|&)"+ name +"=([^&]*)(=&|$)");
    var r = window.location.search.substr(1).match(reg);
    if (r != null) return unescape(r[2]);
    return null;
}
const imageSource = getUrlParam("source");
const originalImage = document.getElementById("image");
origin();

```

选项卡

通过点击切换选项卡功能的实现，使用jquery。

```

$(document).ready(function () {
    const nav = $('#nav').find('li');
    const action = $("#action_container").find('.tab');
    var i = 0;
    nav.each(function () {
        $(this).attr("index", i);
        i++;
        $(this).click(function () {
            reset();
            if($(this).text() === "剪切") {
                create_cropper();
            }
            nav.attr('class', '');
            action.hide();
            $(this).attr('class', 'act');
            action.eq([$($this).attr('index')]).show();
        })
    });
    action.hide();
    nav.first().trigger("click");
})

```

滤镜功能

使用Lena.js的库实现，对其源代码稍作修改以呈现正确的图片大小。

```

function change_filter(type) {
    var filter = LenaJS[type];
    LenaJS.filterImage(filteredImageCanvas, filter, originalImage);
    filteredImage.src = filteredImageCanvas.toDataURL("image/png");
}

```

调整功能

内嵌代码以实现滑动条和输入框的互相更新。

```



使用Caman.js的库，对其源代码稍作修改以避免html的元素替换。


```

```
function adjust() {
    const brightness = element_brightness.val();
    const saturation = element_saturation.val();
    const hue = element_hue.val();
    const test = Caman("#image", function () {
        this.brightness(brightness);
        this.saturation(saturation);
        this.hue(hue);
        this.render(function () {
            filteredImage.src = this.toBase64("png");
        });
    });
}
```

剪切功能

使用Copper.js的库实现。

```
let copper;
function rotate(degree) {
    copper.rotate(degree);
}
function create_copper() {
    copper = new Copper(originalImage, {
        preview: ".preview",
        autoCrop: false,
        ready: function () {
            this.copper.crop();
        }
    });
}
function apply_crop() {
    originalImage.src = copper.getCroppedCanvas().toDataURL("image/png");
    copper.destroy();
}
```